19_Web Application Vulnerability Detection Using Taint Analysis

by Herbertus Yulianton

Submission date: 11-Apr-2023 12:57AM (UTC+0700)
Submission ID: 2060750036
File name: Web_Application_Vulnerability_Detection_Using_Taint_Analysis.pdf (9.57M)
Word count: 1638
Character count: 9257

Web Application Vulnerability Detection Using Taint Analysis and Black-box Testing

Heribertus Yulianton[1,2*], Agung Trisetyarso[1], Wayan Suparta[1,3], Bahtiar Saleh Abbas[1], Chul Ho Kang[1]
[1]Computer Science Department, Binus Graduate Program | Doctor of Computer Science, Bina Nusantara University, West Jakarta, Indonesia
[2]Faculty of Information Technology Universitas Stikubank Semarang
[3]Department of Informatics Universitas Pembangunan Jaya

Email : *heri@edu.unisbank.ac.id

Abstract. Web applications continue to grow however web attacks are also increasing, this shows an increase in web application vulnerabilities. Several methods have been used to detect vulnerabilities in web applications such as black-box testing, dynamic analysis, and static analysis. In this article, we propose a framework for detecting web application vulnerabilities by combining all three methods. The strengths of one method are used to overcome the weaknesses of another method. This framework is believed to provide better results than if each method was used separately.

1. Introduction

Web applications continue to grow therefore web application vulnerabilities become a critical issue. However, a report shows that web attacks are increasing in 2019 [1]. This shows there are still many vulnerabilities in web applications that can be exploited by attackers. Several methods have been used to detect web application vulnerabilities such as black-box testing [2-4], static analysis [5,6], and dynamic analysis [7, 8].

Black-box testing is a popular choice because it is able to operate automatically regardless of the server-side language used [2]. But black-box testing has the disadvantage of potentially not testing hidden pages. Black-box testing works by observing the output of a web application when a web application is given a certain input. Web applications that are quite complex usually have many web pages, which sometimes are not all linked or can be traced from the first page of the web. These pages are referred to as hidden pages.

Static and dynamic analysis is carried out from the server side. Therefore, static and dynamic analysis provides a more complete picture of the application being analyzed. However, static and dynamic analysis gives higher false positive results compared to black-box testing. In this article, we propose a framework that combines all three methods to detect web application vulnerabilities to get the best results. This framework consists of three modules, namely the dynamic taint analysis module, the static taint analysis module, and the black-box testing module. The system first tested will be input into the dynamic taint analysis module and the source code will be input into the static taint analysis module. The results obtained from the two modules will be additional information for the black-box testing module.

Web application vulnerability detection by combining several methods has been done before. Petukhov and Kozlov [9] detected a web application vulnerabilities using the penetration testing module first then the results were inputted into the dynamic analysis module. This was done in two stages. The scheme can be seen in Figure 1.

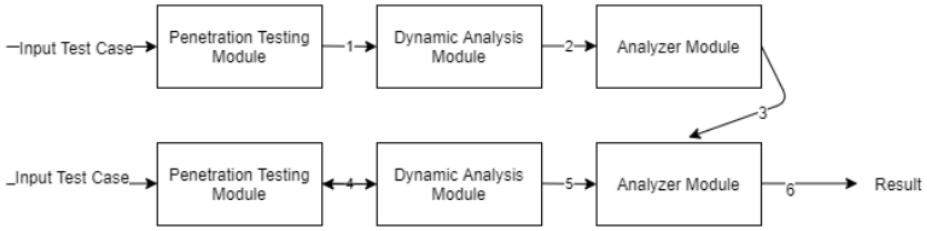

Figure 1. The combined scheme of penetration testing and dynamic analysis approaches [9].

Vogt et al. [10] combine dynamic data tainting and static analysis to prevent cross-site scripting. Zhang et al. [11] created an SDCF framework that combines static and dynamic analysis as can be seen in Figure 2.

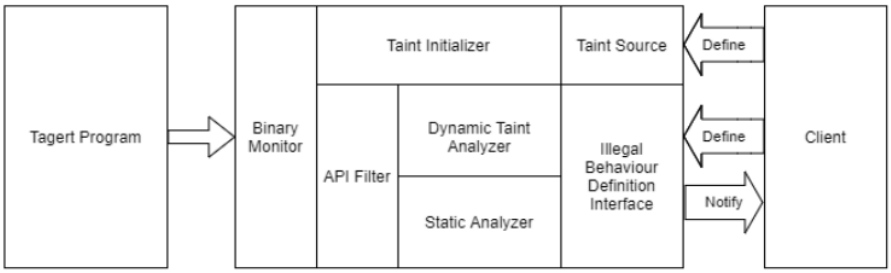

Figure 2. Framework SDCF [11].

2. Method

The proposed framework consists of three modules, namely the dynamic taint analysis module, the statistical taint analysis module, and the black-box testing module. Each module and the integrated module will be explained below.

2.1. Dynamic taint analysis module

Taint analysis is based on the concept that data (such as user input) cannot be trusted. Taint analysis can be divided into two, namely dynamic taint analysis and static taint analysis. Dynamic taint analysis performs analysis at the machine level so that it does not require the availability of source code. Some tools that use this method are: SwordDTA [8], Bitblaze [12], and Dytan [13].

2.2. Static taint analysis module

Static taint analysis module first reads the source code and converts it into Abstract Syntax Tree for further analysis. Some tools that use this method include: TAJ [14], Pixy [5], and Andromeda [15].

2.3. Black-box testing module

Black-box testing or penetration testing automatically performs vulnerability scanning without access to the program code but only through the input form. Many tools are available that use this method both paid and freely available. Vulnerable code like in figure 3 can be detected easily.

```
class VulnerCode
{
   private $hook;

   function __construct()
   {
      // some PHP code...
   }

   function __wakeup()
   {
      if (isset($this->hook)) eval($this->hook);
   }
}

// some PHP code...

$user_data = unserialize($_COOKIE['data']);

// some PHP code...
```

Figure 3. Example of vulnerable code that can be exploited by Code Injection Attack.

2.4. Integrated framework

The dynamic taint analysis module will accept input in the form of binary code from the tested web application. The static taint analysis module will accept input in the form of source code from the web application being tested. Both modules can be run in parallel. The result of both modules is a list of possible vulnerabilities of the web application. Then black-box testing is done on the list. Figure 4 shows the proposed integrated framework.

3. Results and Discussion

The output of the dynamic taint analysis module and static taint analysis module is a list of web application inputs that are suspected of containing vulnerabilities. The results of the taint analysis, as shown by [14], still have large false positives. For this reason, at the final stage of this experiment, we conducted black-box testing by providing certain inputs through a list of inputs generated from the previous stage and analyzing the output results. The results of the analysis are an indication of the vulnerability of the web application.

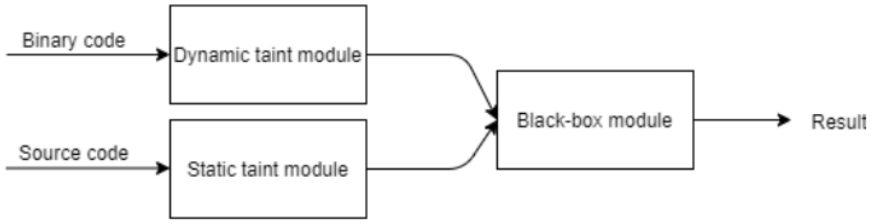

Figure 4. Integrated framework.

4. Conclusion

This article provides a proposed framework for web application vulnerability detection. This framework combines three different methods, namely dynamic analysis, static analysis, and black-box testing using the advantages of one method to overcome the weaknesses of other methods. Combining these three methods is believed to overcome the weaknesses of each method and produce high accuracy and low false positives. This framework consists of three modules, namely the dynamic taint analysis module, the static taint analysis module, and the black-box testing module. The next work is to implement this framework.

Acknowledgement

This research was supported by Universitas Stikubank Semarang. We thank our colleagues from Doctor of Computer Science Department Bina Nusantara University who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

References

[1] Symantec. 2019. Internet Security Threat Report VOLUME 24 Tech. rep. Symantec

[2] Doupé, A., Cavedon, L., Kruegel, C., & Vigna, G. 2012. Enemy of the state: A state-aware black-box web vulnerability scanner. In *Presented as part of the 21st {USENIX} Security Symposium ({USENIX} Security 12)*, pp. 523-538.

[3] Deepa, G., Thilagam, P. S., Khan, F. A., Praseed, A., Pais, A. R., & Palsetia, N. 2018. Black-box detection of XQuery injection and parameter tampering vulnerabilities in web applications. *International Journal of Information Security*, **17**(1), pp. 105-120.

[4] Ceccato, M., Nguyen, C. D., Appelt, D., & Briand, L. C. 2016, September. SOFIA: An automated security oracle for black-box testing of SQL-injection vulnerabilities. In *2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 167-177.

[5] Jovanovic, N., Kruegel, C., & Kirda, E. 2006, May. Pixy: A static analysis tool for detecting web application vulnerabilities. In *2006 IEEE Symposium on Security and Privacy (S&P'06)*, pp. 6-pp.

[6] Yan, X. X., Wang, Q. X., & Ma, H. T. 2017, October. Path sensitive static analysis of taint-style vulnerabilities in PHP code. In *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, pp. 1382-1386.

[7] Karim R, Tip F, Sochurkova A and Sen K. 2019. *IEEE Transactions on Software Engineering,* 1–1.

[8] Cai, J., Zou, P., Ma, J., & He, J. 2016. SwordDTA: A dynamic taint analysis tool for software vulnerability detection. *Wuhan University Journal of Natural Sciences*, **21**(1), pp. 10-20.

[9] Petukhov, A., & Kozlov, D. 2008. Detecting security vulnerabilities in web applications using dynamic analysis with penetration testing. *Computing Systems Lab, Department of Computer Science, Moscow State University*, pp. 1-120.

[10] Vogt, P., Nentwich, F., Jovanovic, N., Kirda, E., Kruegel, C., & Vigna, G. 2007, February. Cross Site Scripting Prevention with Dynamic Data Tainting and Static Analysis. In *NDSS*, **2007**, pp. 12.

[11] Zhang, R., Huang, S., Qi, Z., & Guan, H. 2012. Static program analysis assisted dynamic taint tracking for software vulnerability discovery. *Computers & Mathematics with Applications*, **63**(2), pp. 469-480.

[12] Song, D., Brumley, D., Yin, H., Caballero, J., Jager, I., Kang, M. G., ... & Saxena, P. 2008, December. BitBlaze: A new approach to computer security via binary analysis. In *International Conference on Information Systems Security*, pp. 1-25.

[13] Clause, J., Li, W., & Orso, A. 2007, July. Dytan: a generic dynamic taint analysis framework. In *Proceedings of the 2007 international symposium on Software testing and analysis*, pp. 196-206.

[14] Tripp, O., Pistoia, M., Fink, S. J., Sridharan, M., & Weisman, O. 2009. TAJ: effective taint analysis of web applications. *ACM Sigplan Notices*, **44**(6), pp. 87-97.

[15] Tripp, O., Pistoia, M., Cousot, P., Cousot, R., & Guarnieri, S. 2013, March. Andromeda: Accurate and scalable security analysis of web applications. In *International Conference on Fundamental Approaches to Software Engineering*, pp. 210-225.

19_Web Application Vulnerability Detection Using Taint Analysis

ORIGINALITY REPORT

23%	20%	18%	14%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

MATCH ALL SOURCES (ONLY SELECTED SOURCE PRINTED)

10%

★ S O Putri, Firmansyah. "The Efficiency of Steel Material as Buildings Construction", IOP Conference Series: Materials Science and Engineering, 2020

Publication

Exclude quotes	On	Exclude matches	< 2%
Exclude bibliography	On		